International Journal of Computer Trends and Technology
ISSN: 2231–2803 / https://doi.org/10.14445/22312803/IJCTT-V72I9P126

Volume 72 Issue 9, 165-169, September 2024

Original Article

Model-Based Safety Analysis for Autonomous Driving

Shobhit Kukreti[1], Nidhi Bhardwaj[2]

[1]*Independent Researcher, Carnegie Mellon University, PA, USA.*
[2]*Independent Researcher, San Jose State University, CA, USA.*

[1]*Corresponding Author : skukreti@ieee.org*

Received: 09 August 2024 Revised: 07 September 2024 Accepted: 26 September 2024 Published: 30 September 2024

Abstract *- Self-driving Cars or Autonomous Driving has been a formidable technological hurdle. While the hardware has evolved to produce high-fidelity sensors such as 4k GMSL cameras, LiDars and Radars, mimicking the correct human behavior has proven to be tougher than early expectations. It requires a departure from conventional design, security, and validation procedures to ensure the development of a reliable and secure system. This manuscript delineates the application of a Model-Based Safety Analysis methodology (MBSA) to an Advanced Driver-Assistance System (ADAS) employing a modular numerical simulation platform. We explain the various activities occurring at each stage and delineate the associated objectives. Furthermore, we present experimental simulation outcomes that underscore the advantages inherent in this approach.*

Keywords *- ADAS, Autonomous Driving, MBSA, Vehicle Safety, Simulation.*

1. Introduction

The question of ensuring the dependability and safety of autonomous cars through study and argumentation remains unresolved. While automotive standard ISO 26262 [1] primarily addresses internal faults such as random hardware and systematic software faults, it offers little guidance on managing challenges related to interpreting sensed data and interacting with the environment—crucial aspects of autonomy. Moreover, assessing the gravity of potential events has been based on the control exerted by a human driver, a criterion that has become obsolete in the world of autonomous vehicles. To solve the above challenge, we bring forward an approach, adapting a methodology from aeronautics known as Model-Based Safety Analysis (MBSA), which focuses on environmental perception and interaction issues. MBSA necessitates constructing a system model at a high level of abstraction, enabling the generation of all minimal paths (referred to as cutsets by analogy with failure trees) to a given feared event. Our innovative approach involves utilizing MBSA to identify potentially critical scenarios, namely sets or sequences of internal failures associated with interpretation or interaction problems. While the abstract nature of the model may not cover all critical scenarios of the system, this method is still effective and provides unparalleled assurance. It surpasses the conventional approach of simulating a finite set of scenarios in a detailed model (typically a physical simulator) where the set of possible scenarios is infinite. Each scenario generated by MBSA represents a "class" of scenarios akin to those simulated in a physical simulator. Subsequently, we validate these scenarios in a physical simulator, establishing the precise parameter bounds of the critical scenario class. To demonstrate the efficacy of this method, we apply it to the autonomy function TJC (Traffic Jam Chauffeur/Assist), responsible for navigating the vehicle in a traffic jam scenario while following a lead car at a maximum speed of 70 km/h on the highway. Section II captures an overview of functional safety and our model development in Section III. Section IV presents the results of our approach, encompassing both sequences and cut sets, followed by References in Section V.

2. Modelling the Autonomy Function

We start with modelling the Traffic Jam Chauffeur/Assist (TJC) Controller and all associated elements (e.g., obstacles, leading vehicles) in the language of Model Based Safety Analysis (MBSA) to generate multiple scenarios. We look for scenarios involving issues in the sensing process, such as glare in images or erroneous patterns and image recognition.

2.1. Tools Utilized

We shall use Simfia, a tool developed by Apsys-Airbus for TJC. It is based on the formal language AltaRica [7]. AltaRica was initially conceived in LaBRI4 and offers an easy-to-use interface. An AltaRica model comprises a collection of bricks, each with input and output connectors. Its state is determined by the variables and events. An event causes a change in the values of the state variables, thereby creating a state machine. For example, an event labelled "failure" moves the variable "s" from "nominal" to "failed". The output connector "o" is set to the value "ko" through logical expressions. AltaRica also provides the definition of a brick hierarchy, where a composite brick comprises other

bricks. Starting from an observation point for a hazardous event, the AltaRica compiler generates cutsets, defined as sets of events leading to the abnormal situation. While AltaRica modelling is highly abstract and capable of generating all cutsets, a physical model is significantly more detailed but cannot generate all paths resulting in a given situation. Moreover, the AltaRica compiler can produce sequences—sets of events with a specified occurrence order. For example, considering the model depicted in Figure 1, with a simple failure in each brick:

- Cutsets include {C}, {D}, and {A,B}.
- Sequences encompass (C), (D), (A, B), and (B, A).

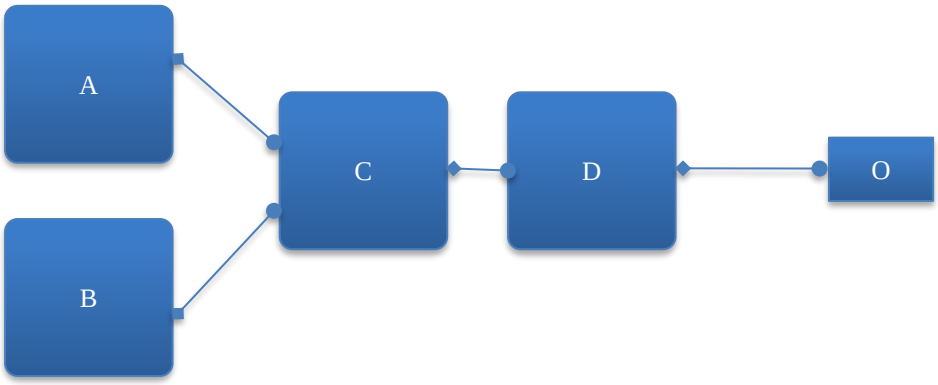

Fig. 1 Model example with 4 bricks

2.2. Model Structure

Our model of the Advanced Driver Assistance System (ADAS) consists of both the self-driving vehicle and the surrounding environment. The environment consists of other vehicles, current weather conditions, etc. The above is required to determine the multiple hazardous scenarios.

The model depicted in Figure 2 consists of the following blocks:

2.2.1. Environment

Comprises of all the relevant environmental input and output elements for the Traffic Jam Chauffeur (TJC)

2.2.2. Perception

Includes sensors or vision of the autonomous vehicle.

2.2.3. Fusion

Involves merging similar classes of information obtained from different sensors e.g. camera and radar.

2.2.4. Control

Handles the decision-making process regarding vehicle motion behavior. It typically involves a CAN controller sending commands to the car's Electronic Control Unit.

2.2.5. Activation Conditions of the Controller

Specifies the conditions under which the driver can activate the TJC.

2.2.6. Environment

The environmental factors include street/road lane markings, weather conditions (sunny, cloud), street signs, and stationary obstacles. Moreover, when considering the changing environment, traffic is accounted for as two other cars in our ego vehicle's vicinity.

2.2.7. Perception

This includes external and internal sensors measuring our vehicle's acceleration, speed and yaw rate. The external sensors are a front camera and a front radar, which provide information about obstacles, other vehicles, lane markings, and street signs.

2.2.8. Fusion

The camera and radar sensors complemented each other. While the camera provides excellent visual data in optimum conditions, the radar data is useful for gathering obstacle data in adverse conditions as well.

The fused data is necessary to provide a crucial and unified view of the surroundings to the Traffic Jam Chauffer/Assist. Data from several sensors, as in the case above, increases confidence levels.

2.2.9. Activation Conditions of the TJC Controller

The controller can be activated by the vehicle driver when they have hands on the steering and have fastened seatbelts.

If drivers fail to meet the above requirements, the system triggers a warning to the driver, followed by emergency braking if necessary.

2.2.10. Control

The system control module manages velocity via Adaptive Cruise Control (ACC) and the vehicle's lateral trajectory via the Lane Keep Assist feature. It has an algorithm to ensure steady vehicle behavior under varying conditions.

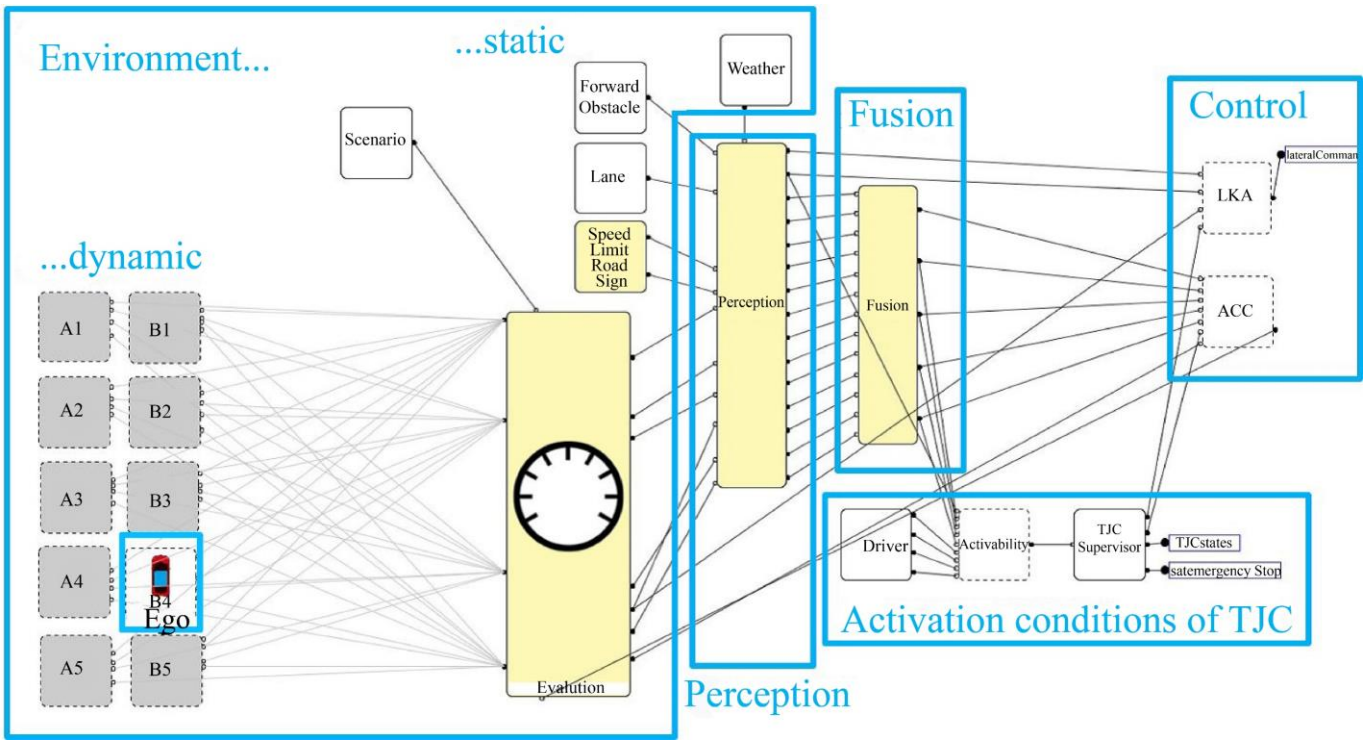

Fig. 2 Proposed model overview

2.3. Modelling Perception

Proper modelling of the simulation, including the vehicle and its perception, is a cornerstone for examining Traffic Jam Assist safety. We propose three data types to model environment elements: flows, objects without parameters, and objects with parameters. They account for various scenarios and probable failures in the perception flow.

Perception errors in autonomous cars are often due to how the software incorrectly categorizes the sensed data or due to adverse environmental interaction rather than hardware failures. We distinguish between these functions to isolate the different types of perception errors and model them separately.

2.4. Modelling Control Consequences and Dynamic Environment

To address dynamic traffic scenarios such as:

2.4.1. Other Vehicles on the Road

This model adds two additional vehicles that can freely change their speed and lane, trying to mimic the actual behavior of other drivers.

We put restrictions so as to avoid unrealistic scenarios so that our ego vehicle in simulation does not violate any real-world scenario.

We limit the number of cars in our simulation to two. In the real world, the number of vehicles around our ego vehicle would be more, but modeling such a system with more than two car dependencies will complicate our model.

2.5. Looping ACC Control Command to Speed Value

This is a closed-loop system to minimize the difference between the set speed and the actual speed. With the dynamic nature of traffic and a dependency on the ego vehicle's behavior, we define a control loop between the Adaptive Cruise Control command and the ego speed (set-speed). The loop ensures that a change in the environment or system state triggers a new command from ACC. This is an event which activates to update the ego speed value.

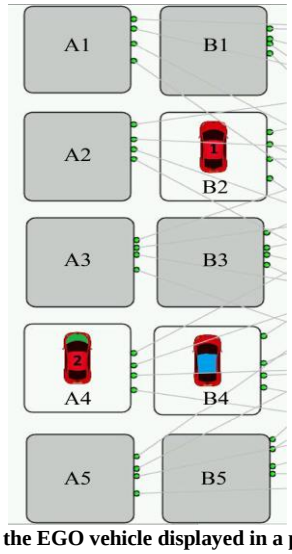

Fig. 3 In blue, the EGO vehicle displayed in a possible scene

2.6. Road Situation Visualization

Multiple display bricks are used to represent the road situation. These bricks create a scene around our ego vehicle, maintaining a constant velocity as the ego vehicle. The visual data helps in debugging and validating the model's behavior through multi-scenario testing while also providing an easy-to-understand view to the consumers of data.

3. Results

This section talks about the process for extracting information from the model using the Simfia tool. We generate both cutsets and sequences, resulting in the hazardous "collision" event. Cutsets are events where order is not important, while sequences are ordered. Both are utilized to provide two levels of interpretation.

3.1. Cutset Analysis

We obtained 60 hours of data with cutsets of order 4, whereas the typical order for industrial studies is order 3. It is shown in table 1. We post-process the cutset data via filtering, removing redundant data, etc. We filter the cutset data where the TJC algorithm has failed and requests the driver to take over the car.

Removing events of position change decided by ACC, as these represent system evolution.

This processing of data leads to 39 cutsets across 6 categories based on their characteristics.

Table 1. Cutsets results

Order	Simfia Generated	Pruned List
1	1	1
2	29	6
3	153	19
4	283	13
Total	466	39

3.2. Sequence Analysis: We enforce that each cut set is present with its permutations in the sequence set. After that, we identify scenarios where incident order matters, highlighting important sequences for further analysis. After extracting critical scenarios, we simulate them using the SCANeR Studio application.

4. Conclusion

The manuscript presents a model-based safety analysis for autonomous vehicles/self-driving vehicles using AltaRica language and ***Simfia*** application. We choose a use-case of Traffic Jam Assist, aiming to improve the model's accuracy without complicating the algorithm.

We generate and analyze sequences and cutsets for the collision event and simulate scenarios. Future work shall involve further increasing the number of surrounding vehicles and modelling cases outside the highway driving scenario to urban driving.

Fig. 4 SCANeR Studio visualization module when simulating

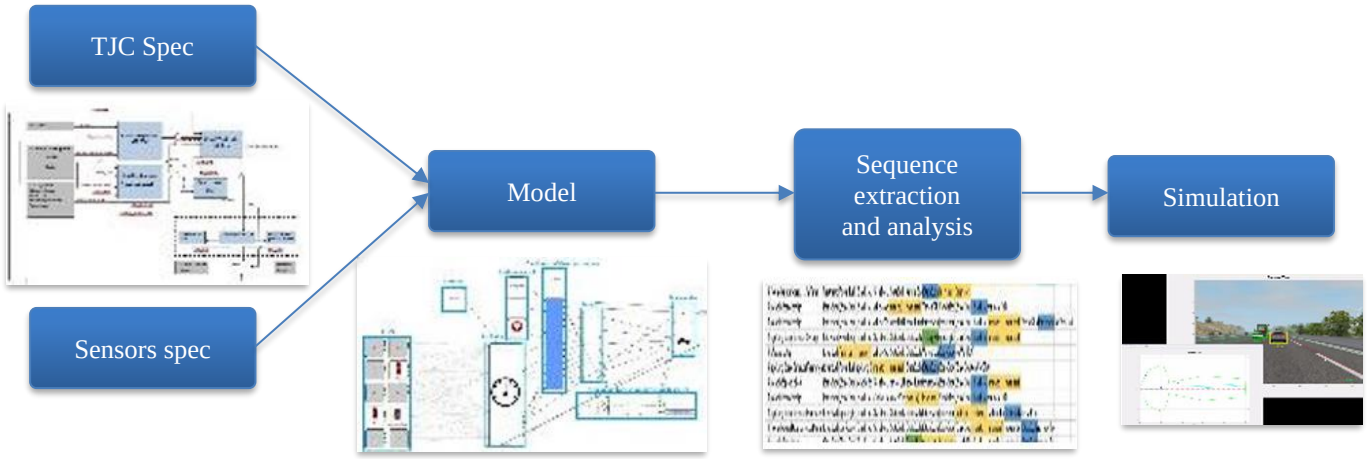

Fig. 5 The MBSA approach

References

[1] ISO 26262-1:2011, "*Road Vehicles - Functional Safety*," ISO Standards, 2011. [Google Scholar] [Publisher Link]

[2] J3016_201609, "*Taxonomy and Definitions for Terms Related to Driving Automation Systems for on-Road Motor Vehicle*," SAE International, 2016. [Google Scholar] [Publisher Link]

[3] ISO 21448:2022, "*Road Vehicles - Safety of the Intended Functionality*," ISO Standards, 2022. [Google Scholar] [Publisher Link]

[4] Brian Paden et al., "A Survey of Motion Planning and Control Techniques for Self-Driving Urban Vehicles," *IEEE Transactions on Intelligent Vehicles*, vol. 1, no. 1, pp. 33-55, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[5] Rowan McAllister et al., "Concrete Problems for Autonomous Vehicle Safety: Advantages of Bayesian Deep Learning," *International Joint Conference on Artificial Intelligence*, pp. 4745-4753, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[6] Pierre Bieber, Charles Castel, and Christel Seguin, "Combination of Fault Tree Analysis and Model Checking for Safety Assessment of Complex System," *4th European Dependable Computing Conference Toulouse*, France, pp. 19-31, 2002. [CrossRef] [Google Scholar] [Publisher Link]

[7] André Arnold et al., "The AltaRica Formalism for Describing Concurrent Systems," *Fundamenta Informaticae*, vol. 40, no. 2-3, pp. 109-124, 1999. [CrossRef] [Google Scholar] [Publisher Link]